Volume 111, Numher 2, March-April 2006
Journal of Research of the National Institute of Standards and Technology

[J. Res. Natl. Inst. Stand. Technol. **111**, 103-111 (2006)]

Optimization Models for Scheduling of Jobs

Volume 111 | Number 2 | March-April 2006

S. H. Sathish Indika and Douglas R. Shier

Department of Mathematical Sciences,
Clemson University,
Clemson, SC 29634-0975

This work is motivated by a particular scheduling prohlem that is faced hy logistics centers that perform aircraft maintenance and modification. Here we concentrate on a single facility (hangar) which is equipped with several work stations (hays). Specifically, a number of johs have already heen scheduled for processing at the facility; the starting times, durations, and work station assignments for these johs are assumed to he known. We are interested in how hest to schedule a number of new johs that the facility will be processing in the near future. We first develop a mixed integer quadratic programming model (MIQP) for this prohlem. Since the exact solution of this MIQP formulation is time consuming, we develop a heuristic procedure, hased on existing hin packing techniques. This heuristic is further enhanced by application of certain local optimality conditions.

Key words: aircraft; bin packing; heuristic; integer programming; maintenance; optimization; scheduling.

Accepted: December 14, 2005

Available online: http://www.nist.gov/jres

1. Introduction

The present study was motivated by a problem encountered by an aircraft logistics center that houses a number of facilities (hangars), each of which contains multiple work stations (bays). The center offers its customers a wide variety of aircraft maintenance services (jobs), including structural repairs, passenger/freight conversions, and engine changes. In this paper we concentrate on a single hangar which is equipped with several bays.

It is assumed that the company has contractual obligations for a number of existing jobs, which have to be completed in the hangar. These existing jobs are characterized by their specified starting times, bay assignments, and job durations (*spans*). Thus, a given bay will be idle during the time periods between existing jobs. We are interested in how best to schedule a number of new jobs into these vacant spaces. Specifically, the company wishes to construct a manufacturing schedule that can be flexible in accommodating future incoming jobs.

First we present a mixed integer quadratic programming (MIQP) approach to model this scheduling problem. Solution of this MIQP gives a schedule that optimizes the overall facility utilization, while providing increased flexibility to accommodate future jobs. Next we develop local optimality conditions for the common types of changes that can occur to a particular schedule (job switches). These conditions enable us to better understand optimal solutions and also to conduct sensitivity analyses.

The exact MIQP formulation of the scheduling problem has mn binary variables, where m denotes the number of vacant spaces and n denotes the number of new jobs. Since there are 2^{mn} possible choices for the binary variables, exact solution of our MIQP model is possible only for very small problems. Consequently we devel-

op a heuristic procedure by considering the vacant spaces as *bins* with varying capacities and applying some modified bin packing techniques. We can also incorporate the local optimality conditions to improve this basic heuristic approach.

2. Optimization Model

First we consider a very simplified scenario in order to motivate our optimization model. Suppose there is only one bay and, relative to the jobs already scheduled for that bay, there are three vacant spaces (bins) B_1, B_2, B_3 of lengths (capacities) 8 d, 12 d, 15 d. Also suppose that three new jobs J_1, J_2, J_3 with spans of 6 d, 7 d, 10 d are to be assigned to the bay. Several ways of assigning the three new jobs to the three bins are depicted in Fig. 1. Intuitively, it seems that assignment A_3 is the most desirable of the three, because it provides the largest contiguous space remaining in a bin (9 d) after assigning the three jobs. For example, by adopting assignment A_3, the center is capable of handling a future job that requires a span of 9 d or handling two additional jobs with spans of 6 d and 3 d, whereas the other two assignments cannot. Here we observe that creating a large amount of space in a bin (or in a few bins) after assigning the jobs is generally preferred over small amounts of space that are scattered over many bins.

To be more precise, let us define the *residual capacity* of a bin as the capacity that remains after assigning a job (or jobs) to that particular bin. Since the sum of the residuals is fixed (in this example totaling 12) for all assignments, we consider the sum of squared residuals for these assignments. Assignment A_1 has residual capacities 8, 2, 2 with sum of squares 64 + 4 + 4 = 72; similarly, A_2 has residual capacities 2, 5, 5 with sum of squares 54, while A_3 has residual capacities 1, 2, 9 with sum of squares 86. We observe that assignment A_3 has the largest sum of squares, as a result of the contribution from the highest residual capacity term. In general, we argue that an assignment with the largest sum of squares should be preferred because it leaves a higher residual capacity in a bin (or bins) instead of smaller residual capacities that are scattered over several bins. In turn this gives added flexibility to the company in accommodating future incoming jobs. We are thus motivated to use the maximization of the sum of squared residuals as our objective criterion. Since the sum of the residuals is fixed, we are equivalently maximizing the variance of the residuals.

2.1 Formulation

To formulate this problem, suppose that n new jobs are to be scheduled using m existing bins, with c_k being the capacity of bin B_k. Also suppose that job J_i has span s_i. Define the binary variable $x_{ik} = 1$ if job J_i is assigned to bin B_k, and $x_{ik} = 0$ otherwise. Also let z_k denote the residual capacity of bin B_k. Then our optimization model for the scheduling problem has the form:

$$\max z = \sum_{k=1}^{m} z_k^2$$

$$\sum_{i=1}^{n} s_i x_{ik} + z_k = c_k, \quad k = 1,..., m$$

$$\sum_{k=1}^{m} x_{ik} = 1, \quad i = 1,..., n$$

$$x_{ik} \in \{0,1\}, \quad z_k \geq 0.$$

The above optimization model has $m + n$ equality constraints involving mn binary variables and m continuous variables. The first m equality constraints define the residual capacity variables z_k; the nonnegativity of z_k ensures that the set of jobs assigned to bin B_k should not exceed that bin's capacity c_k. The remaining n equality constraints require that each job should be

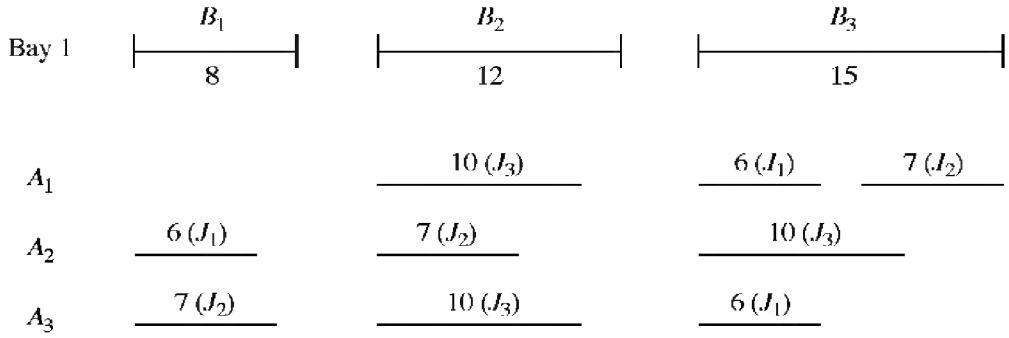

Fig. 1. Assignments A_1, A_2, A_3.

assigned to exactly one bin. Since we are optimizing a quadratic objective function over the (linear) constraint set, this model can be classified as a mixed integer quadratic programming (MIQP) problem. Since the objective function is a sum of squares, this problem is a type of convex maximization problem, a very difficult type of mixed integer optimization problem.

Thus far we have not considered the starting time t_k associated with each bin. We can easily incorporate such starting times into our MIQP model by simply modifying the objective function:

$$\max z(\beta) = \sum_{k=1}^{m} (z_k + \beta t_k)^2, \qquad (1)$$

where β is a nonnegative valued parameter. When $\beta = 0$, we recover our original model in which maximizing the sum of squared residuals ("good fit") is the sole criterion. However, by increasing the value of β, we emphasize the contribution of the starting time and thus encourage new jobs to be assigned earlier in time. The parameter β can be interpreted as the increased importance that the decision maker would like to give to earlier completion of jobs. It is not hard to verify that $z(\beta)$ is a piecewise quadratic, increasing convex function of β.

The above model can be easily applied to situations in which there are multiple bays within a hangar. Indeed for the purposes of the optimization model MIQP, it is only necessary to maintain a list of bins with their capacities and starting times, without regard to the particular bay associated with each bin. In the multiple bay context, the parameter β can still be interpreted as the relative importance of a good fit versus an assignment of jobs to bins occurring earlier in time.

2.2 Examples

In order to get some insights into the proposed MIQP model we present two numerical examples. We have used the global optimization package LINGO [4] to solve these MIQP problems exactly. *Example 1* involves $m = 5$ bins and two bays: the first three bins B_1, B_2, B_3 are associated with bay 1 and the other two bins B_4, B_5 with bay 2. Capacities and starting times for these bins are given in Table 1. Four new jobs J_1, J_2, J_3, J_4 are to be assigned; these jobs have the span times s_i given in Table 2. Using LINGO we obtained the exact solution of the MIQP model for various values of β. It turned out that there were only three sets of optimal solutions as the parameter β was varied. For $0 \le \beta \le 0.11$, the optimal assignment A_1 allocates jobs J_1, J_2, J_3, J_4 to bins B_2, B_4, B_3, B_1 respectively. A schematic illustration of assignment A_1 is given in Fig. 2.

Table 1. Bin capacities and starting times for Example 1

	bin B_k				
k	1	2	3	4	5
c_k	10	12	10	14	16
t_k	6	22	43	4	28

Table 2. Span times of new jobs for Example 1

	job J_i			
i	1	2	3	4
s_i	12	11	9	10

For the range $0.11 \le \beta \le 0.4$, the optimal assignment A_2 allocates jobs J_1, J_2, J_3, J_4 to bins B_4, B_2, B_3, B_1 respectively. Thus the only modification that occurs to the optimal job schedule, in changing from assignment A_1 to assignment A_2, is that jobs J_1, J_2 interchange positions between bins B_2, B_4. Over the final range $0.4 \le \beta < \infty$, the optimal assignment A_3 allocates jobs J_1, J_2, J_3, J_4 to bins B_4, B_2, B_5, B_1 respectively. The only difference between assignment A_2 and assignment A_3 is the movement of job J_3 from bin B_3 to bin B_5. This behavior is consistent with our earlier observation that increasing β

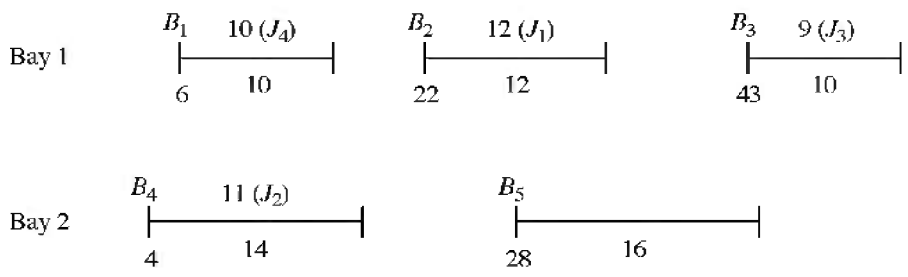

Fig. 2. Assignment A_1, optimal for $0 \le \beta \le 0.11$.

favors moving jobs to bins that occur earlier in time: B_5 has a starting time of 28, earlier than the starting time 43 for B_3.

Example 2 involves $m = 9$ bins and three bays, each containing three bins. We are interested in assigning five new job J_1, ..., J_5 to these bins. The data for this problem are given in Tables 3 and 4. Here we also investigated the changes in optimal solutions as the parameter β is varied. For this problem, there were eight sets of optimal solutions as the parameter β was varied over the ranges: [0, 0.004], [0.004, 0.033], [0.033, 0.05], [0.05, 0.079], [0.079, 0.15], [0.15, 1], [1, 3], [3, ∞]. As one illustration, Fig. 3 presents the optimal solution A_4 over the first of these ranges. In assignment A_4, jobs J_1, J_4, J_5 are assigned to bins B_3, B_4, B_5 whereas jobs J_2, J_3 are assigned to bin B_9.

Table 3. Bin capacities and starting times for Example 2

	bin B_k								
k	1	2	3	4	5	6	7	8	9
c_k	17	20	13	14	16	18	15	19	21
t_k	5	32	63	4	25	52	3	24	51

Table 4. Span times of new jobs for Example 2

	job J_i				
i	1	2	3	4	5
s_i	12	11	10	14	16

3. Local Optimality Conditions

In the previous section, we observed several transformations that occurred in changing from one optimal schedule to another optimal schedule. Specifically, in changing from A_2 to A_3, a job (J_3) moves from one bin to another bin, a transformation we call a *move*. In changing from A_1 to A_2, two jobs (J_1, J_2) interchange positions between two bins, termed a *swap*. In addition, another simple transformation involves transferring jobs from several bins to a common bin (*grouping*) or vice versa (*ungrouping*). Collectively we call these types of changes, and combinations thereof, *job switches*. In this section we will first study the individual effects of each type of job switch upon a given assignment. This enables us to develop local optimality conditions for these types of job switches, relative to a given assignment. Using these local optimality conditions, it is possible to identify job switches that can improve a given feasible solution. Moreover, these local optimality conditions can aid us in carrying out a sensitivity analysis with respect to the parameter β.

3.1 Move

Consider the two assignments A and B shown in Fig. 4. In assignment A, job J_i is currently assigned to bin B_u, whereas in assignment B, job J_i has been moved to bin B_v. Otherwise the two assignments are identical. The starting times of bins B_u and B_v are t_u and t_v respectively, and the span time of job J_i is s_i. Define a_u (resp. a_v) as the capacity utilized by the other jobs that are already assigned to B_u (resp. B_v), and r_u (resp. r_v) as the remaining capacity of bin B_u (resp. B_v) before the

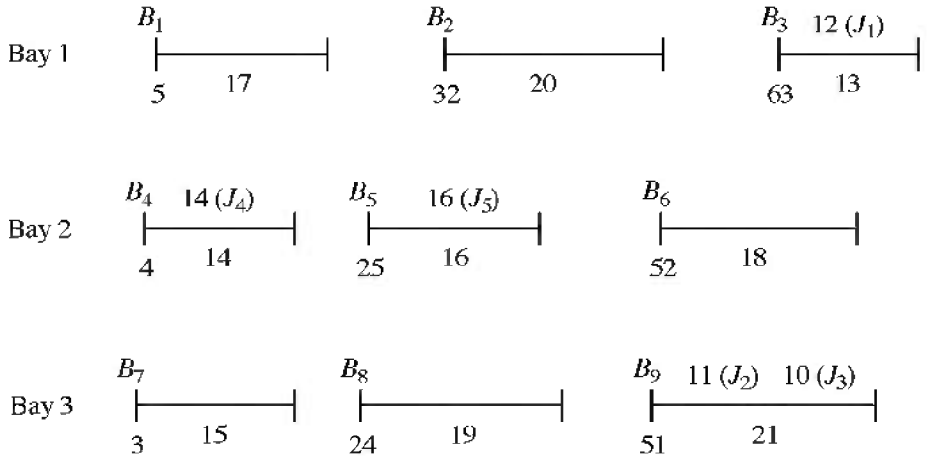

Fig. 3. Assignment A_4, optimal for $0 \le \beta \le 0.004$.

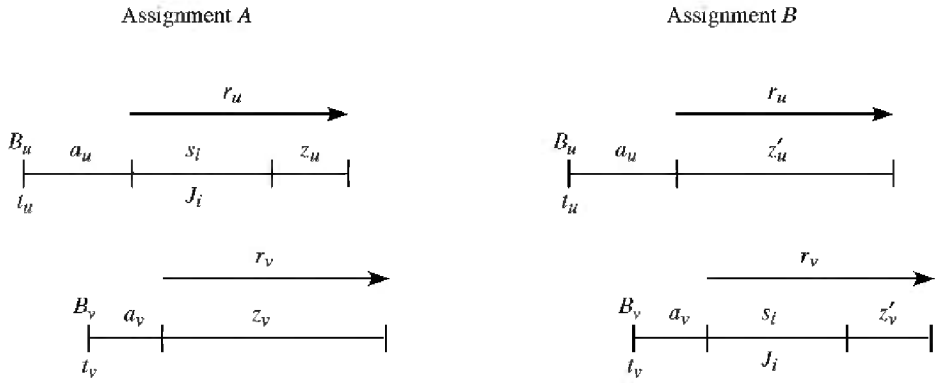

Fig. 4. Job J_i moves from bin B_u to bin B_v.

assignment of job J_i. The residual capacities of bins B_u and B_v in assignment A are defined as z_u and z_v respectively, whereas the corresponding residual capacities of bins B_u and B_v in assignment B are defined as z'_u and z'_v. Thus $z_u = r_u - s_i$, $z_v = r_v$, $z'_u = r_u$, $z'_v = r_v - s_i$.

Let Z_A, Z_B denote the objective function values in Eq. (1) for assignments A, B. Since the only difference between the two assignments is the movement of job J_i from bin B_u to bin B_v, we can ignore the other contributions in the objective function values Z_A and Z_B, writing

$$Z_A = (z_u + \beta t_u)^2 + (z_v + \beta t_v)^2 = (r_u - s_i + \beta t_u)^2 + (r_v + \beta t_v)^2$$

$$Z_B = (z'_u + \beta t_u)^2 + (z'_v + \beta t_v)^2 = (r_u + \beta t_u)^2 + (r_v - s_i + \beta t_v)^2$$

If we define $y_u = r_u + \beta t_u$, $y_v = r_v + \beta t_v$, then the above expressions simplify to

$$Z_A = (y_u - s_i)^2 + y_v^2, \qquad Z_B = y_u^2 + (y_v - s_i)^2.$$

Consequently $\Delta Z = Z_A - Z_B = 2(y_v - y_u)s_i$ is the change in objective function (1). Assignment A will be locally optimal with respect to the move in Fig. 4 if and only if $\Delta Z \geq 0$; since $s_i > 0$ this is equivalent to $y_v - y_u \geq 0$. However, if $y_v - y_u < 0$ then it is beneficial to move job J_i from bin B_u to bin B_v, assuming this move is feasible.

It will be convenient to use an alternative representation to depict the move shown in Fig. 4. Namely, we denote the bins B_u and B_v by vertices u and v, and we use a directed edge i to denote the movement of job J_i from bin B_u to bin B_v. This is depicted in Fig. 5.

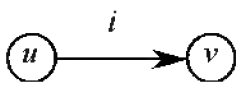

Fig. 5. Graphical representation of a move.

To illustrate how this local optimality condition can be applied, consider Example 1 and assignment A_2, which allocates jobs J_1, J_2, J_3, J_4 to bins B_4, B_2, B_3, B_1. The only feasible moves involve transferring a job from one of the bins $B_1,\ldots, B_4$ to bin B_5. For example, assignment A_2 remains locally optimal for moving job J_3 from bin B_3 to B_5 if and only if $0 \leq y_5 - y_3 = (16 + 28\beta) - (10 + 43\beta) = 6 - 15\beta$, so that $\beta \leq 0.4$. The other conditions $0 \leq y_5 - y_1$, $0 \leq y_5 - y_2$, $0 \leq y_5 - y_4$ hold automatically since $\beta \geq 0$. The result is that assignment A_2 is locally optimal under moves for all $\beta \leq 0.4$, consistent with the results obtained earlier from LINGO.

3.2 Swap

We now consider the swapping of jobs, relative to two assignments A and B. In assignment A, jobs J_i and J_k are assigned to bins B_u and B_v respectively, whereas in assignment B, jobs J_i and J_k are assigned to bins B_v and B_u. Therefore the only difference between the two assignments is that jobs J_i and J_k are swapped between bins B_u and B_v. This is illustrated in Fig. 6. Again define a_u (resp. a_v) as the capacity utilized by the other jobs that are assigned to B_u (resp. B_v), and r_u (resp. r_v) as the remaining capacity of bin B_u (resp. B_v) before the assignment of jobs J_i and J_k. The residual capacities of bins B_u and B_v in assignment A are defined as z_u and z_v respectively, whereas the corresponding residual capacities of bins B_u and B_v in assignment B are defined as z'_u and z'_v. Thus $z_u = r_u - s_i$, $z_v = r_v - s_k$, $z'_u = r_u - s_k$, $z'_v = r_v - s_i$.

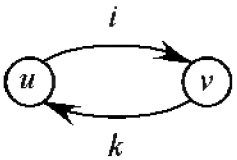

Fig. 6. A swap of jobs J_i and J_k.

As before, let Z_A, Z_B be the objective function values for assignments A, B. Since the only difference between the two assignments is the swapping of jobs J_i and J_k between bins B_u and B_v, we can ignore the other contributions in the objective function values Z_A and Z_B, writing

$$Z_A = (z_u + \beta t_u)^2 + (z_v + \beta t_v)^2 = (r_u - s_i + \beta t_u)^2$$
$$+ (r_v - s_k + \beta t_v)^2 = (y_u - s_i)^2 + (y_v - s_k)^2$$
$$Z_B = (z'_u + \beta t_u)^2 + (z'_v + \beta t_v)^2 = (r_u - s_k + \beta t_u)^2$$
$$+ (r_v - s_i + \beta t_v)^2 = (y_u - s_k)^2 + (y_v - s_i)^2$$

Therefore $\Delta Z = Z_A - Z_B = 2(y_v - y_u)s_i + 2(y_u - y_v)s_k = 2(y_v - y_u)(s_i - s_k)$ and assignment A is locally optimal with respect to the indicated swap if and only if $\Delta Z \geq 0$. Otherwise it is beneficial to swap jobs J_i and J_k, assuming that such a swap is feasible.

By way of illustration, consider assignment A_1 of Example 1, shown in Fig. 2. The only feasible swaps involve exchanging jobs J_1 and J_2 or exchanging jobs J_3 and J_4. Assignment A_1 remains locally optimal for swapping job J_1 in bin B_2 with job J_2 in B_4 if and only if $(y_4 - y_2)(s_1 - s_2) \geq 0$. Simplifying produces $0 \leq [(14 + 4\beta) - (12 + 22\beta)](12 - 11) = 2 - 18\beta$ or $\beta \leq \frac{1}{9}$. Swapping jobs J_3 and J_4 between bins B_3 and B_1 imposes the condition $(y_3 - y_1)(s_4 - s_3) \geq 0$ which simplifies to $37\beta \geq 0$, which clearly holds for $\beta \geq 0$. In summary, assignment A_1 is locally optimal for swaps over the range $0 \leq \beta \leq \frac{1}{9}$, consistent with the results obtained earlier from LINGO.

3.3 Grouping and Ungrouping

Another simple job switch involves the grouping of jobs, in which p jobs are moved from separate bins to a new bin. Specifically, suppose that in assignment A, jobs $J_1, \ldots, J_p$ are assigned to bins $B_{u_1}, \ldots, B_{u_p}$, whereas in assignment B, they are all grouped together in the single bin B_v. This transformation is illustrated in Fig. 7. Here we can define the utilized capacity of all bins after the assignment of jobs other than $J_1, \ldots, J_p$, as well as the residual capacities of bins $B_{u_1}, \ldots, B_{u_p}$, in a manner similar to that done previously.

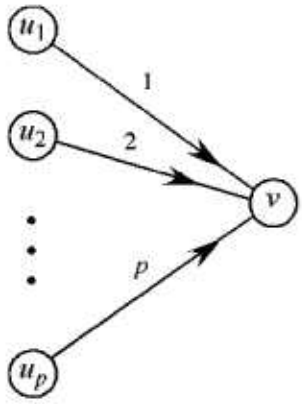

Fig. 7. Grouping of jobs $J_1, \ldots, J_p$.

The associated objective function values for these assignments are given by

$$Z_A = \sum_{i=1}^{p} (z_{u_i} + \beta t_{u_i})^2 + (z_v + \beta t_v)^2 = \sum_{i=1}^{p} (r_{u_i} - s_i + \beta t_{u_i})^2$$
$$+ (r_v + \beta t_v)^2$$
$$Z_B = \sum_{i=1}^{p} (z'_{u_i} + \beta t_{u_i})^2 + (z'_v + \beta t_v)^2 = \sum_{i=1}^{p} (r_{u_i} + \beta t_{u_i})^2$$
$$+ (r_v - \sum_{i=1}^{p} s_i + \beta t_v)^2$$

Simplification then produces

$$\Delta Z = Z_A - Z_B = 2\left[\sum_{i=1}^{p} (y_v - y_{u_i}) s_i - \sum_{i<j} s_i s_j\right].$$

An analogous development produces the following expression when p jobs $J_1, \ldots, J_p$ are moved from a single bin B_u to p separate bins $B_{v_1}, \ldots, B_{v_p}$:

$$\Delta Z = Z_A - Z_B = 2\left[\sum_{i=1}^{p} (y_{v_i} - y_u) s_i + \sum_{i<j} s_i s_j\right].$$

Notice the change in sign in the last summation from negative to positive for the ungrouping ΔZ, compared with the grouping ΔZ.

To illustrate the ungrouping of jobs, consider assignment A_4 in Example 2, shown in Fig. 3. Suppose that jobs J_2, J_3 are moved from bin B_9 to bins B_7, B_1 respectively. (This is a feasible ungrouping of jobs.) Then

$$\Delta Z = 2[(y_7 - y_9)s_2 + (y_1 - y_9)s_3 + s_2 s_3]$$
$$= 2([(15 + 3\beta) - (21 + 51\beta)]11 + [(17 + 5\beta)$$
$$- (21 + 51\beta)]10 + 11 \cdot 10) = 2(4 - 988\beta).$$

Consequently, the proposed ungrouping will be advantageous when $\Delta Z < 0$ or $\beta > \frac{1}{247}$.

3.4 More Complex Job Switches

This section considers combining the previous transformations (moves, swaps, groupings, and ungroupings) into more complex job switches. To motivate the general case, we first consider an example involving eight jobs and five bins. Namely, the transformations that occur, in changing from the current assignment A to the new assignment B are as follows: jobs J_1, J_5 swap between bins B_a, B_c; jobs J_2, J_3 ungroup from bin B_a to bins B_d, B_e; job J_4 moves from B_b to B_c while job J_6 moves from B_d to B_b; and jobs J_7, J_8 ungroup from B_e to B_b, B_c. See Fig. 8 for a graphical representation of this more complex rearrangement. Following the previous development, we obtain

$$Z_A = (y_a - s_1 - s_2 - s_3)^2 + (y_b - s_4)^2 + (y_c - s_5)^2 + (y_d - s_6)^2 + (y_e - s_7 - s_8)^2$$

$$Z_B = (y_a - s_5)^2 + (y_b - s_6 - s_7)^2 + (y_c - s_1 - s_4 - s_8)^2 + (y_d - s_2)^2 + (y_e - s_3)^2$$

$$\Delta Z = 2[(y_c - y_a)s_1 + (y_d - y_a)s_2 + (y_e - y_a)s_3 + (y_c - y_b)s_4 + (y_a - y_c)s_5 + (y_b - y_d)s_6 + (y_b - y_e)s_7 + (y_c - y_e)s_8 + s_1s_2 + s_2s_3 + s_1s_3 + s_7s_8 - s_1s_4 - s_4s_8 - s_1s_8 - s_6s_7].$$

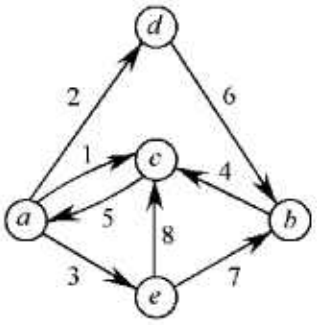

Fig. 8. Graphical representation of a more complicated scenario.

Notice that for each directed edge (i, j) representing the movement of job J_k in Fig. 8 there is a term $2(y_j - y_i)s_k$ in ΔZ. There are also terms in ΔZ to represent the ungrouping and grouping of jobs. Jobs J_1, J_2, J_3 ungroup at vertex a, giving rise to the product terms $s_1s_2 + s_2s_3 + s_1s_3$; likewise the ungrouping of jobs J_7, J_8 at vertex e gives the term s_7s_8. On the other hand, jobs J_1, J_4, J_8 group at vertex c, producing the (negative) term $-(s_1s_4 + s_4s_8 + s_1s_8)$, while jobs J_6, J_7 group at vertex b, giving $-s_6s_7$.

In general suppose that the directed graph $G = (V, E)$ represents the specified changes to a current assignment, where the set V of vertices represents the bins and the set E of directed edges represents the movement of jobs between bins. Denote the span time of the job associated with edge (i, j) by s_{ij}. Also let $\Gamma^+(i)$ denote the set of edges leaving vertex i and let $\Gamma^-(i)$ denote the set of edges entering vertex i. Then the expression for ΔZ becomes

$$\Delta Z = 2\sum_{(i,j)\in E}(y_j - y_i)s_{ij} + \sum_{i\in V}\sum\{s_a s_b : a, b \in \Gamma^+(i), a \neq b\} - \sum_{i\in V}\sum\{s_a s_b : a, b \in \Gamma^-(i), a \neq b\}.$$

4. Heuristic Procedures

In Sec. 2.1, we modeled the scheduling problem as an MIQP with mn binary variables and m continuous variables, where m is the number of bins and n is the number of new jobs. Since the number of possible choices 2^{mn} for the binary variables rapidly becomes large, even for small m and n, the exact mathematical solution of the MIQP model is very time consuming. Therefore we consider heuristic solution approaches in this section, rather than exact procedures.

As noted earlier the vacant time interval between any two existing jobs can be considered a bin. Since the intervals between existing jobs are of different length, the corresponding bins have variable size. The underlying problem is then to assign the new jobs to these bins in such a way that the set of new jobs assigned to a bin fits within that bin's capacity; this is to be done in an "optimal" fashion. Thus our scheduling problem can be viewed as a variable-sized bin packing problem [3,5] with an unusual type of objective function. In this section we will first develop a simple heuristic procedure based on existing bin packing techniques. Next we will apply the local optimality conditions developed in Sec. 3 to improve this bin packing heuristic.

4.1 Bin Packing Heuristic

We begin by reviewing two bin packing algorithms that are well known in the literature [2]. They are the First Fit (FF) algorithm and the Best Fit (BF) algorithm. The objective of such standard bin packing algorithms is to minimize the number of bins that are need-

ed to pack a given set of items. The FF algorithm assigns the next job into the lowest indexed bin into which it will fit. If the next job does not fit into any existing bin then we open a new bin and place the next job in the new bin. The BF algorithm is similar to the FF algorithm, except that it assigns the next job into that bin which will leave the smallest residual capacity after the assignment.

In our heuristic algorithm we first order the jobs in order of nonincreasing span times. There is an intuitive appeal to ordering the span times in this manner: we assign the jobs with higher span times first and hope that we can accommodate the jobs with smaller span times using the spaces that remain. A similar strategy is adopted in existing bin packing algorithms [1,6]. Before assigning the next job (in order of nonincreasing span) to a bin, we first identify the bins into which the job can fit. Among these candidate bins, we select a bin, say bin B_u, having the minimum value of $z_u + \beta t_u$. As shown in the proposition below, this procedure enables us to (locally) improve the current solution as much as possible at the next step. Next we assign the new job to the selected bin and update the residual capacity of that bin. We follow this procedure until all jobs have been assigned. We now summarize more formally the steps of this heuristic algorithm.

bin_packing

1. Order the jobs by nonincreasing span time.
2. For the next job J_k in order, with span s_k, consider only those bins with residual capacity at least s_k. Among such bins, select bin B_u to minimize $z_u + \beta t_u$.
3. Assign job J_k to bin B_u and update $z_u \leftarrow z_u - s_k$.
4. If there are additional jobs to be processed, go to Step 2.

Here $\beta \geq 0$ is the same parameter introduced in Sec. 2.1. When $\beta = 0$ we assign the next job J_k into a bin (into which it fits) having the minimum current residual capacity. Since this will also leave the smallest residual capacity after the assignment, our heuristic algorithm is similar to the BF algorithm for $\beta = 0$. When β is large, we assign jobs to bins in order of increasing starting time. In other words, jobs are assigned to bins that occur earliest in time. This is analogous to assigning a job to the lowest indexed bin into which it will fit, when bins are ordered by starting times. So when β is large, our heuristic algorithm behaves similar to the FF algorithm. In this way we have blended both BF and FF into our particular bin packing heuristic. We now demonstrate that our heuristic performs the locally "best" assignment for the current job at each step of the algorithm.

Proposition. *The bin packing heuristic locally improves the objective function by as much as possible at the next step.*

Proof. Suppose that job J_k with span s_k is to be assigned and that bins $B_1, ..., B_p$ are the bins into which J_k can fit. Let $z_1, ..., z_p$ be the current residual capacities of these bins. Select bin B_u such that

$$z_u + \beta t_u = \min\{z_1 + \beta t_1, ..., z_p + \beta t_p\}. \qquad (2)$$

Let Z_j be the objective function value (1) obtained by assigning job J_k to bin B_j at the next step. We claim that $Z_u \geq Z_j$ for all $1 \leq j \leq p$. Let α denote the contribution to the objective function from all bins other than B_u and B_j in the current assignment. Thus

$$Z_u = \alpha + (z_u - s_k + \beta t_u)^2 + (z_j + \beta t_j)^2$$

$$Z_j = \alpha + (z_u + \beta t_u)^2 + (z_j - s_k + \beta t_j)^2$$

giving

$$Z_u - Z_j = -2z_u s_k - 2\beta t_u s_k + 2z_j s_k + 2\beta t_j s_k$$

$$= 2[(z_j + \beta t_j) - (z_u + \beta t_u)]s_k \geq 0,$$

where the final inequality follows from Eq. (2). □

We illustrate the bin packing heuristic using Example 1 when $\beta = 0.3$. Ordering the four jobs by nonincreasing span times places them in the sequence J_1, J_2, J_4, J_3. To begin, job J_1 can fit into bins B_2, B_4, B_5, which have residual capacities 12, 14, 16 and starting times 22, 4, 28. Since $\min\{12 + 0.3(22), 14 + 0.3(4), 16 + 0.3(28)\} = 15.2$ is achieved for bin B_4, we assign job J_1 to bin B_4 and update the residual capacity of bin B_4 to $z_4 = 2$.

We next select (in order) job J_2, which can fit into bins B_2, B_5, with residual capacities 12, 16. Since $\min\{12 + 0.3(22), 16 + 0.3(28)\} = 18.6$ is achieved for bin B_2, we assign job J_2 to bin B_2 and update the residual capacity of bin B_2 to $z_2 = 1$. Continuing in this fashion, job J_4 is assigned to bin B_1 and job J_3 is assigned to bin B_3. Thus the heuristic assigns jobs J_1, J_2, J_3, J_4 to bins B_4, B_2, B_3, B_1 respectively. This assignment is identical to the optimal assignment A_2 found for the range $0.11 \leq \beta \leq 0.4$. In fact for Example 1 the heuris-

tic produces optimal assignments over all three ranges for the parameter β.

For Example 2 the MIQP model gives eight optimal assignments corresponding to eight ranges of the parameter β. The assignments obtained by the heuristic procedure are identical to the optimal assignments obtained by the MIQP model in seven of these eight ranges. The only difference occurs over the range $0 \le \beta \le 0.004$ for which the optimal assignment A_4 is shown in Fig. 3; it assigns jobs J_1, ..., J_5 to bins B_3, B_9, B_9, B_4, B_5. By contrast, the heuristic procedure obtains a different assignment A_4', in which J_1, ..., J_5 are assigned to B_3, B_7, B_1, B_4, B_5. If however we perform in A_4' a grouping of jobs J_2, J_3 from bins B_7, B_1 to bin B_9, then the change in objective function value is

$$\Delta Z = 2[(y_9 - y_7)s_2 + (y_9 - y_1)s_3 - s_2 s_3] = 2(-4 + 988\beta).$$

For $0 \le \beta < 0.004$, the term in parentheses above is negative so that it is advantageous to perform this grouping. In other words, first applying the heuristic to obtain A_4' and then using an improving step (grouping) does indeed yield the optimal assignment A_4 over the range.

To summarize, for Example 1 the heuristic procedure obtained the optimal solution for all ranges. In Example 2, there was one instance in which the heuristic gave a suboptimal solution. However in this case, a single job switch (grouping) was sufficient to produce the optimal solution. This encouraging success suggests a hybrid heuristic procedure that first carries out the bin packing algorithm, followed by local improvements using selected job switches. In particular, it is straightforward to check for improving moves and swaps; it is a bit more tedious to evaluate all groupings and ungroupings relative to a given assignment.

5. Conclusions

In this paper we have considered the scheduling of different types of aircraft maintenance programs. In this initial study, we concentrated on the simplified case where there is only a single hangar. First we developed an MIQP to model this scheduling problem. The MIQP model incorporates a parameter β that reflects the relative importance of a good fit versus assigning of jobs to bins occurring earlier in time. For small test problems, it is possible to obtain an exact schedule by solving the MIQP model. Results from these test problems indicated that there were relatively few optimal schedules over the range of all possible values of the specified parameter β. We also developed local optimality conditions for certain types of job switches, relative to a given assignment. The local optimality conditions enable us to improve a given feasible job schedule. Because exact solution of the MIQP is limited to fairly small instances, we developed a simplified heuristic procedure based on existing bin packing techniques. An area for future research is to combine the bin packing heuristic with the intelligent application of the local optimality conditions.

Acknowledgments

We are pleased to dedicate this paper to Christoph Witzgall, whose insights into modeling and optimization have been constant sources of inspiration.

6. References

[1] S. Chopra and D. Simchi-Levi, Bin packing, in Handbook of Discrete and Combinatorial Mathematics, K. H. Rosen, ed., CRC Press, Boca Raton (2000) pp. 1001-1003.
[2] E. G. Coffman, Jr., M. R. Garey, and D. S. Johnson, Approximation algorithms for bin packing: an updated survey, in Algorithm Design for Computer System Design, G. Ausiello, M. Lucertini, and P. Serafini, eds., Springer, Vienna (1984) pp. 49-106.
[3] D. K. Friesen and M. A. Langston, Variable sized bin packing, SIAM J. Comput. **15**, 222-230 (1986).
[4] LINGO User's Guide, LINDO Systems Inc., Chicago (2003).
[5] F. D. Murgolo, An efficient approximation scheme for variable-sized bin packing, SIAM J. Comput. **16**, 149-161 (1987).
[6] P. Schwerin and G. Wäscher, The bin-packing problem: a problem generator and some numerical experiments with FFD packing and MTP, Int. Trans. Opl. Res. **4**, 377-389 (1997).

About the authors: *Dr. Douglas Shier was a mathematician in the Center for Applied Mathematics at the National Bureau of Standards (now NIST), from 1975-1980. Currently he is a professor in the Mathematical Sciences Department, Clemson University. Sathish Indika is a graduate student in the computational operations research program at the College of William and Mary.*